

REMARKS

Claims 1-16 remain for consideration.

The Office Action does not establish that claims 1 and 16 are unpatentable under 35 U.S.C. 103(a) over Dessloch, U.S. Patent No. 6,505,211 (“Dessloch”). The rejection is respectfully traversed because the Office Action fails to show that all the limitations are suggested by the Dessloch, fails to provide a proper motivation for modifying the teachings of Dessloch, and fails to show that the modification could be made with a reasonable likelihood of success.

The cited portions of Dessloch do not suggest the limitations in the claims. Furthermore, the Office Action does not address various limitations. The claimed method clearly sets out that various definitions are set forth in program source code, and the compilation of this source code results in the allocating of memory objects and assigning of data values. Even though various keywords of Dessloch are cited to support the allegation that Dessloch suggests the invention, Dessloch’s Abstract teaches otherwise. Specifically, Dessloch’s Abstract teaches:

A method, apparatus, and article of manufacture for providing for persistence of JavaTM objects. A JavaTM object is instantiated from its corresponding JavaTM class definition and then loaded into a JavaTM virtual machine. The class definition corresponding to the JavaTM object can be derived using either the JavaTM Reflection API. Once the class definition is derived, it can be used to inspect the contents of the JavaTM object. A structured type instance is then generated from the inspected contents of the JavaTM object, wherein the structured type instance is stored in a column of a table of a relational database managed by a relational database management system. As a result of these steps, the JavaTM object is persistently stored in the database, yet the persistence semantics for storing the object are not specified as part of the class definition of the object, which means that the persistence semantics are orthogonal to the class definition.

Dessloch derives a class definition of an object using an API call. The claim limitations clearly set out that the first structure type for storage of one or more data values is defined in source code along with a plurality of objects of the structure type. Furthermore, a first class is defined in the source code, with the first class deriving *from* the structure type, the class including a static method configured to convert an object of the structure type to an instance of the class in response to a reference to the method. Those skilled in the art will recognize that the specification in source code of one class deriving from a structure specification is clearly

different from deriving a class definition of an object using an API. Thus, Dessloch's teachings do not suggest the limitations of related to defining the various elements in source code and having the allocation and assignment done at compile time.

The cited portions of Dessloch do not suggest the claim limitations. For example, the limitations of defining in the source program a plurality of objects of the structure type clearly sets out that the definitions are in the source code. The elements of Dessloch alleged in the Office Action appear to teach the opposite. That is, "a FROM_SQL built-in transform of the RDBMS 116 constructs an ObjectStream BLOB (Binary Large Object) that represents a structured type instance 124, and passes it to the JDBC driver 112 (Step 300)." (col. 6, ll. 26-35). Thus, Dessloch's ObjectStream BLOB is constructed during execution, not defined in the source code as claimed.

Nothing in Dessloch is cited as suggesting that the claimed structure type is defined in the same source code program as are the definitions of the objects of the structure type.

Also, in the claimed invention the data values with which the objects of the structure type are initialized are in the same source code program. The Office Action simply cites initialization. There is no apparent suggestion by Dessloch that the objects are initialized with selected data values in the source program.

The limitations further include defining in the source program a first class that derives from the structure type, the class including a static method configured to convert an object of the structure type to an instance of the class in response to a reference to the method. The Office Action is mistaken in alleging that Dessloch teaches these limitations. The Office Action generally refers to a "Java class definition" without providing any particular reference to a specific class. The Office Action goes on to allege that Dessloch's readSQL method satisfies the claim limitations. However, there is no apparent teaching that Dessloch's readSQL is included in any particular class that derives *from* the structure type.

The Office Action further fails to show a suggestion of the limitations of compile-time allocation and initialization. The cited FIGs. 3A, 3B, 6A, and 6B, along with col. 6, ll. 14-58 and col. 9, ll. 21-49 appear to suggest various run-time operations. There is no apparent suggestion of allocation and initialization or of performing these functions at compile-time.

The alleged motivation for modifying Dessloch is that “it would be obvious ... that the java class derives from the structure type since it derives the attributes of the structure type.” This alleged motivation is improper because it appears to misread Dessloch’s teachings. That is, Dessloch teaches deriving the class definition for purposes of inspecting an object (Abstract). This is clearly distinct from and not suggestive of the claimed class deriving from the structure type as defined in the source code. Thus, the alleged motivation is unfounded and improper.

The rejection of claims 1 and 16 over the Dessloch reference should be withdrawn because the Office Action fails to show all the limitations are suggested by the combination, fails to provide a proper motivation for modifying Dessloch, and fails to show that the combination could be made with a reasonable likelihood of success.


Dessloch neither shows nor suggests the claimed invention. Withdrawal of the rejection and reconsideration of the claims are respectfully requested. If the examiner has any questions or concerns, a telephone call to the undersigned is welcome.

The allowability of claims 2-25 is acknowledged. No amendment is made because claims 1 and 16 are also thought to be allowable. Furthermore, The Examiner’s statement for allowance implies that the claimed invention was allowed because the prior art did not disclose certain limitations found in the claims. The limitations characterized by the Examiner, however, if indeed found in the prior art, would not render the claimed invention invalid under 35 USC §102 because the claimed invention includes a number of limitations not addressed in the reasons for allowance. With respect to 35 USC §103, the rigors of establishing a *prima facie* case of obviousness include not only a showing that the prior art teaches the entire claimed invention (all limitations are to be considered), but also that combining the various prior art references is suggested in the art or that there would be motivation to make the combination. Unless Applicants hear otherwise, the comments herein are, as intended, clarifying in a manner consistent with the law.

No extension of time is believed to be necessary for consideration of this response. However, if an extension of time is required, please consider this a petition for a sufficient number of months for consideration of this response. If there are any additional fees in connection with this response, please charge Deposit Account No. 50-0996 (USYS.023PA).

Respectfully submitted,

CRAWFORD MAUNU PLLC
1270 Northland Drive, Suite 390
Saint Paul, MN 55120
(651) 686-6633

By: 
Name: LeRoy D. Maunu
Reg. No.: 35,274